



Introducción a Unidad 3



Daniel Millán & Nicolás Muzi

Mayo 2019, San Rafael, Argentina



UNCUYO
UNIVERSIDAD
NACIONAL DE CUYO



FACULTAD DE
**CIENCIAS APLICADAS
A LA INDUSTRIA**



Editor *vi/vim* y navegación en red

Los temas que se cubrirán son:

1. Introducción a *vi/vim*.
2. Mover y copiar texto en *vi/vim*.
3. Buscar y reemplazar texto en *vi/vim*.
4. Otros comandos útiles en *vi/vim*.
5. Guía rápida de comandos en *vi/vim*.
6. Otros editores Unix: “*emacs*”.
7. Conexión a máquinas remotas.
8. Comandos útiles en rutas de red.
9. Transferencia de archivos a distancia.
10. Otras utilidades relacionadas con Internet.
11. Información de usuario y comunicación en red.
12. Control de impresora.



Objetivos

- Esta Unidad presenta a uno de los editores más populares de UNIX: **vi** (otros son **nano**, **emacs** y **nedit**).
- Para el editor **vi**, en este curso abarcaremos:
 - Introducción de texto básico y navegación.
 - Mover y copiar texto.
 - Buscar y reemplazar texto.
- Además se desarrollarán los principales pasos para navegación y/o trabajo en red (servidor/cluster).
- Para ello se incursionará en:
 - Conexión a equipos remotos.
 - Transferencia de archivos a distancia.
 - Impresión desde línea de comandos.



1. Introducción a *vi/vim*

- **vi**: pronunciado “*vee-eye*”, abreviatura de “*visual*”, o tal vez “*vile*”
- Es un editor de texto basado en el editor de líneas de texto por comandos **ex** (**EX**tended, 1976) basado a su vez en el arcaico **ed** (Thompson, 1971).
- Los principiantes de Unix suelen encontrar **vi** incómodo de usar, no obstante existen razones para que este sea mundialmente empleado:
 - Se encuentra en “**casi**” todo SO tipo Unix, de forma que conocer rudimentos de **vi** es una salvaguarda ante operaciones de emergencia en diversos SOs.
 - Emplea las teclas alfanuméricas para ejecutar órdenes, por lo que se puede utilizar en casi cualquier terminal o estación de trabajo sin tener que preocuparse acerca de las asignaciones de teclado inusuales (ñ,ç,~,€,...).
 - Debido a que utiliza muy pocos recursos suele ser empleado frecuentemente por administradores de sistemas, así como por usuarios que se conectan de forma remota o para tareas simples de edición de archivos de texto.

\$vi *nombreamarchivo* ↵

(abre o crea el archivo *nombreamarchivo*)



1. Introducción a *vi/vim*

- Existe una versión “mejorada” llamada **vim** (**VI iMproved**, 1991)
 - Corrector ortográfico integrado
 - Autocompletado de texto
 - Ventanas múltiples, que dividen el área de edición horizontal o verticalmente
 - Resaltado de sintaxis dependiente del lenguaje de programación (*awk,bash,C,...*)
 - Órdenes deshacer y rehacer
 - Completado de órdenes, palabras y nombres de ficheros
 - Compresión y descompresión de ficheros, que posibilita editar archivos comprimidos
 - Reconocimiento de formatos de fichero y conversión entre los mismos
 - Historial de órdenes ejecutadas
 - Guardado de la configuración entre sesiones
 - Altamente configurable y personalizable
 - Casi 100% compatible con *vi*, pero sin muchos de sus defectos
 - etc ...



1. Introducción a *vi/vim*

- La principal característica que hace único *vi* como editor es su operación basada en modos.
- *vi* tiene dos modos:
 - modo de órdenes: los caracteres que se escriben realizan acciones. Por ejemplo, mover el cursor, cortar o copiar texto, etc.
 - modo de entrada: los caracteres que se escriben se insertan o sobrescriben texto existente.
- Cuando se inicia *vi*, este se encuentra en modo de órdenes.
- Para poner *vi* en el modo de entrada, pulse *i* (*insert*). Ahora puede escribir texto en la posición del cursor; puede corregir errores con la tecla de retroceso a medida que teclee.
- Otra forma de insertar texto, especialmente útil cuando se está al final de una línea es presionar *a* (*append*).
- Para volver al modo de órdenes, presione la tecla **ESC** (*escape*).



1. Introducción a *vi/vim*

- En modo de órdenes, es posible mover el cursor por el documento: **h** (izquierda), **j** (abajo), **k** (arriba) y **l** (derecha). Si se tiene “suerte”, las flechas también suelen funcionar.
- Otras teclas útiles:
 - **^** principio de línea, y **\$** final de línea.
 - **w** principio de la siguiente palabra, **b** comienzo de la palabra anterior.
 - Para ir a la **i**-ésima línea del documento pulse **i** y luego **G** (**5G** le lleva a la línea 5).
 - Para ir al final del documento teclee **G**.
 - Para ir a la siguiente página pulse **^F**, y para volver una página pulse **^B**.
 - Para eliminar texto: mueva el cursor sobre el primer carácter del grupo que desea eliminar y pulse **x** para borrar el carácter actual, **dw** para borrar hasta la palabra siguiente, **d4w** para borrar las próximos 4 palabras, **dd** para borrar la línea, **4dd** elimina 4 líneas incluida la línea donde está el cursor, **d\$** para borrar hasta el final de la línea o incluso **dG** para borrar hasta el final del documento.
 - Para deshacer el último cambio presione **u**.
 - Para unir dos líneas consecutivas pulse **J** (no pulse la tecla de retroceso en el comienzo de la segunda línea!)



2. Mover y copiar texto en *vi/vim*

- *vi* utiliza *buffers* (espacio de memoria) para almacenar el texto que se elimina.
- Hay nueve *buffers* numerados (1-9), así como uno para el comando deshacer.
- Por lo general, *buffer* 1 contiene el texto recientemente eliminado, *buffer* 2 el anterior, etc.
- Para cortar y pegar en *vi*, elimine el texto (p. ej., usando **5dd** borra 5 líneas). A continuación, en la línea que desea pegar el texto pulse **p**.
- Si elimina algo de más, antes de pegar, puede recuperar el texto borrado pegando el contenido de los *buffers*. Puede hacerlo escribiendo "**1p**", "**2p**", etc.
- Para copiar y pegar, "yank" el texto (p. ej., usando **5yy** para copiar 5 líneas). A continuación, en la que línea que desea que aparezca el texto pulse **p**.



3. Buscar y reemplazar texto en *vi/vim*

- En el modo de órdenes, puede buscar un texto especificando expresiones regulares.
- Para buscar hacia delante, escriba / y luego el texto y ↵.
- Para buscar hacia atrás, pulsar ? y luego el texto y ↵.
- Para encontrar el siguiente texto que coincida con la expresión teclee n.
- Para buscar y reemplazar todas las apariciones de “*patrón1*” con “*patrón2*”, escriba:
`%s/pattern1/patrón2/g` ↵ (g puede no ser necesaria en un único archivo)
- Para ser preguntado en cada reemplazo, utilizar c en lugar de g.
- En lugar de % también se puede dar un rango de líneas p. ej.: 5,15s → realiza el reemplazo desde la línea 5 a la 15.



4. Otros comandos útiles en *vi/vim*

- Los programadores pueden gustar de la orden **:set number**↵ que muestra los números de línea (**:set nonumber**↵ quita los número de línea).
- Para guardar un archivo, escriba **:w**↵
- Para guardar y salir, escriba **:wq**↵ o pulse **ZZ**↵
- Para forzar la salida sin guardar, tipee **:q!**↵
- Para empezar a editar otro archivo, tipee **:nombre del archivo**.
- Para ejecutar comandos de *shell* dentro de *vi*, y luego volver a *vi*, tipee **!:shellcommand**↵
- Por ejemplo el carácter % indica el nombre del archivo que está editando, así **!:echo %**↵ imprime el nombre del archivo actual en la terminal.
- . Repite la última orden.



5. Guía rápida en *vi/vim*

Inserting and typing text:

i insert text (and enter input mode)
\$a append text (to end of line)
ESC re-enter command mode
J join lines

Cursor movement:

h left
j down
k up
l right
^ beginning of line
\$ end of line
1G top of document
G end of document
nG go to line *n*
^F page forward
^B page backward
w word forwards
b word backwards
r replace a character below cursor

Deleting and moving text:

Backspace delete character before cursor
 (only works in input mode)
x delete character under cursor
dw delete word
dd delete line (restore with **p** or **P**)
ndd delete *n* lines
d\$ delete to end of line
dG delete to end of file
yy yank/copy line (restore with **p** or **P**)
nyy yank/copy *n* lines

Search and replace:

%s/search string/replace string/g

Miscellaneous:

u undo
:w save file
:wq save file and quit
:q! quit without saving
n move to next word/file



6. Otros editores Unix: “emacs”

- **emacs** es un editor de texto con una gran cantidad de funciones, muy popular entre programadores y usuarios técnicos (Unix, Win, Mac).
- **emacs**: Editor **MACroS** para el TECO (60's) desarrollado por Richard Stallman en el MIT desde 1975, GNU Emacs 1984 (es el más popular) y finalmente aparece **Xemacs** en 1991.
- No es una utilidad estándar de UNIX, pero está disponible en la FSF (*Free Software Foundation*, creada por R. Stallman en 1985 – GNU).
- Un fanático de **emacs** le dirá que proporciona utilidades avanzadas que van más allá de la simple inserción y eliminación de texto. Por ejemplo puede ver dos o más archivos al mismo tiempo, compilar y depurar programas en casi cualquier lenguaje de programación, editar documentos, ejecutar comandos de *shell*, leer páginas del manual, acceder al correo electrónico e incluso navegar por la web, etc...
- Utiliza muchos más recursos que **vi** y sus comandos son rocambolescos.
- **emacs** según los fanáticos de **vi** significa **Escape-Meta-Alt-Control-Shift**.
- En la práctica, la mayoría de los usuarios tienden a utilizar ambos editores *humm*.¹²



7. Conexión a máquinas remotas

- **telnet** (*Telecommunication Network*) es un protocolo/programa de red de los 60s inseguro para conectarse por terminal a máquinas remotas.
 - Todos los datos (incluyendo su nombre de usuario y contraseña) se transmiten por la red en texto plano (sin cifrar).
 - Por esta razón, el acceso **telnet** está desactivado en la mayoría de los sistemas y su uso se debe evitar. El puerto de acceso suele ser el 80.

```
$telnet www.uncu.edu.ar 80↵
```

- **rlogin** (*Remote Login*) es un programa/protocolo para conectarse a máquinas remotas de forma insegura.
- **rsh** (*Remote SHell*) es una aplicación que se basa en el protocolo de **rlogin**, mediante el demonio *rloginid* puede lanzar una *shell* y ejecutar órdenes en máquinas remotas.
- **telnet, rlogin, rsh transmiten información sin cifrar → SSH**



7. Conexión a máquinas remotas

- **SSH (Secure SHell, intérprete de órdenes seguro)** es un protocolo/programa cuya primer versión se remonta a 1995 que permite acceder por terminal de forma segura a máquinas remotas.
- Permite manejar por completo la computadora mediante un intérprete de órdenes.
- Puede redirigir el tráfico de las X (Sistema de Ventanas X) para poder ejecutar programas en un entorno gráfico siempre que se tenga ejecutando un Servidor X (en sistemas Unix y Windows).
- Además de la conexión a otros dispositivos, **SSH** nos permite copiar datos de forma segura (tanto archivos sueltos como simular sesiones **FTP** cifradas), gestionar claves **RSA** para no escribir claves al conectar a los dispositivos y pasar los datos de cualquier otra aplicación por un canal seguro tunelizado mediante **SSH**.

```
$ssh user@servername:~/ ↵
```

```
$ssh user@serverIP:~/ ↵
```



7. Conexión a máquinas remotas

Ejercicio: **SSH** (*Secure SHell*, intérprete de órdenes seguro) conéctese al servidor de mecánica en el nodo informático de la FCAI-UNCuyo.

```
$ssh alumnoN@192.168.200.2:~/ ↵
```

¿Qué características técnicas posee el servidor de mecánica?

Hints:

CPU: **\$ cat** /proc/cpuinfo

Memory: **\$ free**

\$ cat /proc/meminfo

HDD: **\$ df** -h



8. Comandos útiles en rutas de red

- **ping** es una utilidad de diagnóstico para comprobar el estado velocidad y calidad de una red determinada mediante el envío de paquetes ICMP (*Internet Control Message Protocol*) entre el *host* local y máquinas remotas.

```
$ping www.fcai.uncuyo.edu.ar↵
```

- mide el tiempo de respuesta entre la máquina actual y el servidor web de la FCAI/UNCuyo.
- **ping** también es útil para verificar si una máquina sigue “viva”.

- **tracert** muestra la ruta completa que se recorre hasta llegar a una máquina remota, incluyendo el retraso por cada máquina a lo largo de la ruta. Esto es particularmente útil en la búsqueda de problemas de la red.

```
$tracert www.fcai.uncuyo.edu.ar↵
```




9. Transferencia de archivos en red

- **ftp** (*File Transfer Protocol*) es una forma insegura de transferir archivos entre ordenadores.
 - Cuando se conecta a una máquina a través de **FTP**, se le pedirá su nombre de usuario y contraseña. Puede ingresar su usuario o “*ftp*” o “*anonymous*”.
 - Una vez conectado a través de **FTP**, puede listar los archivos (*dir*), recibir archivos (*get*, *mget*) y enviar archivos (*put*, *mput*). *help* le mostrará una lista de los comandos disponibles. Particularmente útiles son los comandos *binary* (archivos de transferencia preservando los 8 bits) y *prompt n* (no pide confirmar cada archivo en múltiples transferencias de archivos).
 - Escriba *quit* para salir de **ftp** y volver al intérprete de comandos.
- **scp** (*Secure CoPy*) es una manera segura de transferir archivos entre ordenadores. Funciona igual que el comando **cp** de Unix excepto que los argumentos pueden especificar un usuario y la dirección de la máquina, así como los archivos/directorios.

put → `$ scp archlocal user@maquinaremota:archremoto ↵`

get ← `$ scp user@maquinaremota:archremoto . ↵`



10. Otras utilidades en Internet

- **wget** es una aplicación que permite descargar archivos mediante una *shell* desde servidores web (protocolos HTTP, HTTPS, FTP).
 - **wget** no es interactivo, lo que significa que puede funcionar en segundo plano (a diferencia de la mayoría de los navegadores web).
 - Las descargas realizadas mediante **wget** se almacenan como texto HTML puro (se puede ver usando un navegador web).
- **netstat** (*Network Statistics*) muestra el protocolo en uso, las tablas de ruteo, las estadísticas de las interfaces y el estado de la conexión.
- **ifconfig** (*InterFace CONFIGuration*) permite configurar o desplegar numerosos parámetros de las interfaces de red (*eth0*, *eth1*, etc), como la dirección IP (dinámica o estática), o la máscara de red.
- **w3m** es un potente navegador web basado en texto así como un paginador. Permite ser cargado desde *emacs*.
- **netscape**, **lynx** son navegadores web arcaicos y casi en desuso...
- **Nota: w3m, wget, netscape, lynx no son utilidades estándar de Unix.**



11. Información de usuario y comunicación en red

- **finger, who** muestran la lista de los usuarios conectados a una máquina, el terminal que están utilizando, y la fecha en que se conectaron.
- **write** utilizado por los usuarios en una misma máquina que quieren hablar entre sí. Se debe especificar el usuario y (opcionalmente) la terminal en la que están

```
$write alumnos ttys001↵
```

```
hola usuario alumnos↵
```

- Las líneas se transmiten únicamente cuando se presiona ↵.
- Para volver a la línea de comandos, pulse Ctrl-d.

- **talk** es un cliente de chat interactivo más sofisticado que se puede utilizar entre máquinas remotas.

```
$talk alumnos@maquinaremota↵
```

- En la actualidad es muy poco probable que funcione este *chat* debido a los protocolos de seguridad existentes (**ytalk** es otra posibilidad).



12. Control de Impresora

- **lpr** añade un documento a una cola de impresión, por lo que el documento se imprime cuando la impresora está disponible. Mira / *etc/printcap* para averiguar qué impresoras están disponibles.

```
$lpr -P printqueue archivo↵
```

- **lpq** comprueba el estado de la cola de impresión especificada. Cada trabajo tendrá un número de trabajo asociado.

```
$lpq -P printqueue↵
```

- **lprm** elimina la tarea desde la cola de impresión especificada.

```
$lprm -P printqueue jobnumber↵
```

- Nota: tenga en cuenta que **lpr**, **lpq** y **lprm** son utilidades de gestión de impresión al estilo **BSD**. Si está utilizando un estricto **SYSV**, puede que tenga que utilizar los comandos **SYSV** equivalentes: **lp**, **lpstat** y **cancel**.